AMENDMENTS TO THE CLAIMS

1. (Original) In a computer including an I/O device, a method comprising using a virtual machine monitor to commence virtualization of the I/O device at runtime.

2. (Original) The method of claim 1, wherein the computer further includes a CPU, wherein the virtual machine monitor is in control of the CPU prior to the runtime virtualization of the I/O device.

3. (Original) The method of claim 1, wherein the virtualization is performed transparently to the operating system.

4. (Original) The method of claim 1, wherein the I/O device is compatible with the virtualized I/O device.

5. (Original) The method of claim 1, wherein the virtualization includes commencing I/O device emulation at runtime.

6. (Original) The method of claim 5, further comprising configuring the hardware to trap I/O accesses, and enabling the virtual machine monitor to emulate the I/O device in response to the traps.

7. (Currently amended) The method of claim [[5]] 6, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

8. (Original) The method of claim 5, wherein the virtual machine monitor can commence the emulation between I/O sequences.

9. (Currently amended) The method of claim 8, wherein the virtual machine monitor commences emulation by intercepting I/O accesses; wherein the virtual machine monitor uses the intercepted I/O accesses to ~~change the state of~~ update a state machine, whereby the state machine reflects [[the]] a state of the I/O device; and wherein the virtual machine monitor examines

transitions in ~~the state of~~ the state machine to determine whether the I/O device is in the middle of an I/O sequence.

10. (Original) The method of claim 5, wherein the virtual machine monitor can commence the emulation in the middle of an I/O sequence.

11. (Original) The method of claim 5, wherein the virtual machine monitor uses a state machine to determine whether the I/O device is in the middle of an I/O sequence, and delays commencing emulation until the state machine indicates that I/O sequence has completed.

12. (Original) The method of claim 1, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

13. (Currently amended) The method of claim 1, wherein I/O device interrupts are directed to [[the]] an operating system prior to the runtime virtualization of the I/O device; and wherein the I/O device interrupts are directed to the virtual machine monitor during and after the virtualization of the I/O device.

14. (Original) The method of claim 1, wherein the virtual machine monitor temporarily pauses an I/O sequence by emulating the I/O device as being busy.

15. (Original) The method of claim 1, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing virtualization; and wherein the virtual machine monitor restores the determined mode of the operation after virtualization.

16. (Original) The method of claim 1, further comprising devirtualizing the I/O device at runtime following the runtime virtualization.

17. (Original) In a computer including hardware, a virtual machine monitor running on the hardware, an operating system running on the virtual machine monitor, the hardware including an I/O device, the I/O device already

virtualized by the virtual machine monitor, a method comprising devirtualizing the I/O device at runtime.

18. (Original) The method of claim 17, wherein the devirtualization is performed transparently to the operating system.

19. (Original) The method of claim 17, wherein the devirtualization includes stopping I/O device emulation at runtime.

20. (Original) The method of claim 17, wherein the virtual machine monitor emulates the I/O device prior to devirtualization; and wherein the devirtualization includes allowing the virtual machine monitor to temporarily stop the operating system from commencing a new I/O sequence.

21. (Original) The method of claim 20, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

22. (Original) The method of claim 20, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

23. (Currently amended) The method of claim 20, wherein the [[VMM]] virtual machine monitor logs I/O accesses by the operating system to the I/O device during devirtualization, and replays the log to the device after devirtualization, whereby the I/O accesses by the operating system are deferred during the devirtualization of the I/O device.

24. (Original) The method of claim 17, wherein the virtual machine monitor waits for I/Os initiated by the virtual machine monitor's driver for the I/O device to complete, and for all expected interrupts from the device to arrive, before ceasing device emulation.

25. (Original) The method of claim 17, further comprising re-directing interrupts from interrupt handlers in the virtual machine monitor to interrupt handlers in the operating system.

26. (Original) The method of claim 17, further comprising configuring the hardware so the accesses by the operating system to the I/O device no longer trap to the virtual machine monitor.

27. (Original) The method of claim 17, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing devirtualization; and wherein the virtual machine monitor restores the determined mode of the operation after devirtualization.

28. (Original) The method of claim 17, wherein the I/O device is virtualized at runtime again after having been devirtualized at runtime.

29. (Original) A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine for running on the hardware and commencing virtualization of the I/O device at runtime.

30. (Original) The computer of claim 29, wherein the I/O device is compatible with the virtualized I/O device.

31. (Original) The computer of claim 29, wherein the virtualization includes commencing I/O device emulation at runtime.

32. (Original) The computer of claim 31, further comprising configuring the hardware to trap I/O accesses, and enabling the virtual machine monitor to emulate the I/O device in response to the traps.

33. (Original) The computer of claim 32, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

34. (Original) The computer of claim 31, wherein the virtual machine monitor can commence the emulation in the middle of an I/O sequence.

35. (Original) The computer of claim 34, wherein the virtual machine monitor uses a state machine to determine whether the I/O device is in the middle of an I/O sequence, and delays commencing emulation until the state machine indicates that I/O sequence has completed.

36. (Original) The computer of claim 31, wherein the virtual machine monitor temporarily pauses the I/O sequence by emulating the I/O device as being busy.

37. (Original) The computer of claim 29, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

38. (Original) A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine monitor for devirtualizing the I/O device at runtime.

39. (Original) The computer of claim 38, wherein the virtual machine monitor emulates the I/O device prior to commencing devirtualization; and wherein the virtual machine commences the devirtualization by temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

40. (Original) The computer of claim 39, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

41. (Original) The computer of claim 39, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

42. (Original) The computer of claim 39, wherein the virtual machine monitor logs I/O accesses by an operating system to the I/O device during devirtualization, and replays the log to the device after devirtualization.

43. (Original) The computer of claim 39, wherein the virtual machine monitor waits for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

44. (Original) The computer of claim 38, further comprising configuring the hardware so operating system accesses to the I/O device no longer trap to the virtual machine monitor.

45. (Original) The computer of claim 38, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor determines the mode of the I/O device prior to commencing devirtualization; and wherein the virtual machine monitor restores the determined mode of the operation after the I/O device has been devirtualized.

46. (Original) The computer of claim 38, wherein the virtual machine monitor can virtualize the I/O device after having devirtualized the I/O device at runtime.

47. (Currently amended) An article for a computer including an I/O device, the article comprising computer-readable memory encoded with software for causing the computer to commence ~~commencing~~ virtualization of the I/O device at runtime.

48. (Original) The article of claim 47, wherein the virtualization includes commencing I/O device emulation at runtime.

49. (Original) The article of claim 48, wherein the software includes a virtual machine monitor; and wherein the software configures the hardware to trap I/O accesses, and enables the virtual machine monitor to emulate the I/O device in response to the traps.

50. (Currently amended) The article of claim 49, wherein the ~~software includes a~~ virtual machine monitor ~~for using~~ uses memory management to trap the I/O accesses.

51. (Original) The article of claim 48, wherein the software includes a virtual machine monitor for commencing the emulation in the middle of an I/O sequence.

52. (Original) The article of claim 51, wherein the virtual machine monitor includes a state machine for determining whether the I/O device is in the middle of an I/O sequence, the virtual machine monitor delaying the commencement of the emulation until the state machine indicates that the I/O sequence has completed.

53. (Original) The article of claim 52, wherein the virtual machine monitor temporarily pauses the I/O sequence by emulating the I/O device as being busy.

54. (Original) The article of claim 47, wherein the software includes a virtual machine monitor for emulating I/O device interrupts during the runtime virtualization.

55. (Original) The article of claim 47, wherein the software includes a virtual machine monitor for commencing the virtualization of the I/O device at runtime.

56. (Currently amended) An article for a computer including an I/O device, the article comprising computer-readable memory encoded with software for causing the computer to devirtualize ~~devirtualizing~~ the I/O device at runtime.

57. (Original) The article of claim 56, wherein the devirtualization includes ceasing emulation of the I/O device at runtime.

58. (Original) The article of claim 57, wherein the software includes a virtual machine monitor; and wherein the devirtualization includes temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

59. (Original) The article of claim 58, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

60. (Original) The article of claim 58, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

61. (Currently amended) The ~~computer~~ article of claim 57, wherein the software includes a virtual machine monitor for ceasing the emulation; the virtual machine monitor waiting for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

62. (Original) The article of claim 56, wherein the software includes a virtual machine monitor for logging I/O accesses by an operating system to the I/O device during devirtualization, and replaying the log to the I/O device after devirtualization.

63. (Original) The article of claim 56, wherein the software includes a virtual machine monitor, the software configuring the hardware so operating system accesses to the I/O device do not trap to the virtual machine monitor.

64. (Original) The article of claim 56, wherein the I/O device has multiple modes of operations; and wherein the software includes a virtual machine monitor for determining the mode of the I/O device prior to commencing devirtualization; and restoring the determined mode of the operation after the I/O device has been devirtualized.

65. (Original) The article of claim 56, wherein the software includes a virtual machine monitor for devirtualizing the I/O device at runtime.

66. (Original) The article of claim 65, wherein the virtual machine monitor can virtualize the I/O device after having devirtualized the I/O device at runtime.